Training Algorithms - HAM2

Albin Heimerson, Adam Dahlgren

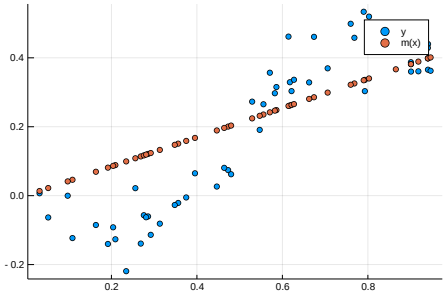
June 9, 2019

As per given instructions, our answers to the given questions are brief but sweet!

1

- $$\begin{aligned} f(\omega) &= \frac{1}{2}\|X^T\omega - y\|^2 \\ &= \frac{1}{2}\omega^T X X^T \omega - y^T X^T \omega + \frac{1}{2} y^T y \\ \nabla f(\omega) &= X X^T \omega - y^T X^T \\ \|\nabla f(x) - \nabla f(y)\| &= \|X X^T (x - y)\| \leq \|X X^T\| \|x - y\| \end{aligned}$$

- Predicted and real y



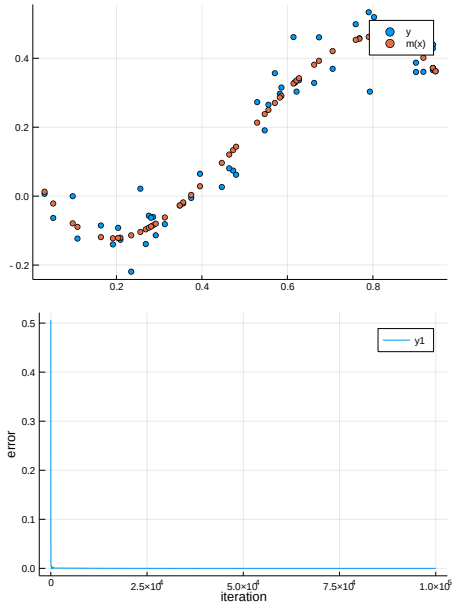
- $$RMSE(m(x)) = 0.1413348190936498$$

2

- Looks like a third order polynomial.

$$X_{poly} = [X^3 \ X^2 \ X \ 1]$$

- $$RMSE(m(x)) = 0.053552052407861636$$



- The result is improved, both a better RMSE and upon visually inspecting the fit we see that it much better captures the data.
- Looking at the data it does not look like it's overfitting, it follows the major structure of the data without focusing on what looks like noise.

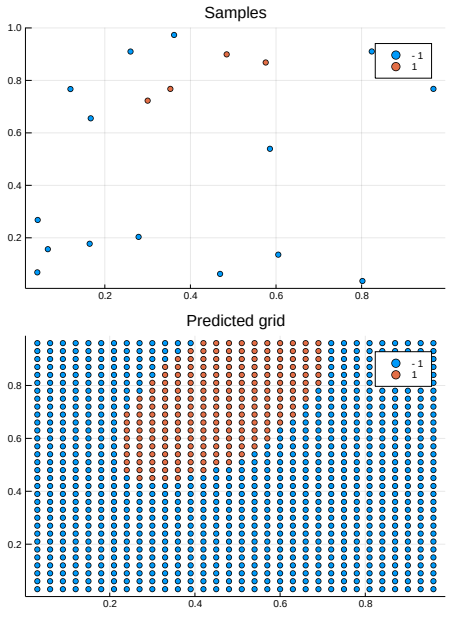
3

- Minimizing the negative likelihood maximizes the likelihood, i.e. maximizes the probability of predicting the given labels from the given data

with the model.

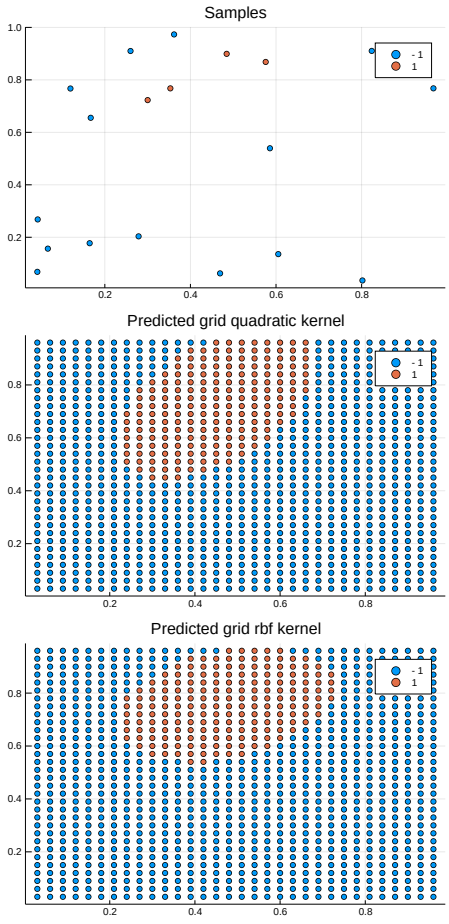
$$
\begin{aligned}
-\log\left(\prod_{i=1}^{n} p_{y_i}(x_i)\right) &= -\sum_{i=1}^{n} \log p_{y_i}(x_i) \\
&= -\sum_{i=1}^{n} \log \frac{1}{1+e^{-y_i \omega^T x_i}} \\
&= \sum_{i=1}^{n} \log(1+e^{-y_i \omega^T x_i}) \\
\min_{\omega}\left(-\log\left(\prod_{i=1}^{n} p_{y_i}(x_i)\right)\right) &= \min_{\omega} \sum_{i=1}^{n} \log(1+e^{-y_i \omega^T x_i})
\end{aligned}
$$

- The classification accuracy is approximately 0.92 and error is 0.08 with around twice as many false negatives as positives.
- The area is maybe slightly bigger than our guess would have been but it's not many points to go on. Increasing the sampled points did not increase the accuracy by much which suggests that the model is not overfitting. But with that said it's still only around 0.94 so there might be room for improvement of the model depending on how much noise there is in the samples.



4

- The quadratic kernel does not seem to overfit the data, but the rbf kernel might be overfitting slightly (we see a reduction in correct predictions for longer training).
- Looking at the samples when we draw more it seems like the noise will make it impossible to get a model with perfect generalization.
- Correct predictions are around 92 % for the quadratic kernel and around 90 % for the rbf kernel (depending on training iterations), and when increasing the number of points both go up towards around 94 % before the calculations take too long to run.



5

We do not think this will improve anything since what is not correctly classified is most likely due to noise in the model. Taking a peek at the file it seems to be some linear transforms and a norm which will result in an ellipsoid. This is then placed in the set depending on some sigmoid probability of the norm which means we will have noise and will never be able to generalize perfectly. Running a simulation of the system and counting the number of points that are labeled correctly according to the defined shape we get that after a hundred million iterations a fraction of 0.94315522 are correct, i.e. we have already reached the limit of what we can hope to achieve in generalization with our previous models.